Data Structures I : Linked lists

Mauricio Toro
Department of Systems and Informatics
Universidad EAFIT

Cocktail of the day: Alexander

Disclaimer: Keep alcohol out of the hands of minors.

Cocktail of the day: Alexander

- 30 ml Cognac
- 30 ml Crème de Cacao
- 30 ml Fresh cream

Word processors use doubly-linked lists

https://www.youtube.com/watch?v=3ViMWbHV_cs

Review: O notation

1. Number of instructions $T(n)$
2. Asintotic notation O
3. Homogeneous lineal recurrence equations

Advantages of Arrays

Figure: Taken from Inc. [?]

Liabilities of Arrays

Liabilities of Arrays

- Arrays have certain disadvantages as data storage structures
  - In an unordered array, searching is slow,
  - In an ordered array, insertion is slow.
  - In both kinds of arrays, deletion is slow.
  - The size of an array cannot be changed.

Taken from [?].

Liabilities of Arrays

- Arrays have certain disadvantages as data storage structures
  - In an unordered array, searching is slow,
  - In an ordered array, insertion is slow.
  - In both kinds of arrays, deletion is slow.
  - The size of an array cannot be changed.

Taken from [?].

Liabilities of Arrays

- Arrays have certain disadvantages as data storage structures
  - In an unordered array, searching is slow,
  - In an ordered array, insertion is slow.
  - In both kinds of arrays, deletion is slow.
  - The size of an array cannot be changed.

Taken from [?].

Liabilities of Arrays

- Arrays have certain disadvantages as data storage structures
  - In an unordered array, searching is slow,
  - In an ordered array, insertion is slow.
  - In both kinds of arrays, deletion is slow.
  - The size of an array cannot be changed.

Taken from [?].

Relationship, not Position

- In an array each item occupies a particular position.
  - This position can be directly accessed using an index number.
    - It is like a row of houses:
    - you can find a particular house using its address.
- In a list the only way to find a particular element is to follow along the chain of elements.
  - It is more like human relations.
    - Maybe you ask Harry where Bob is.
    - Harry does not know, but he thinks Jane might know,
    - so you go and ask Jane...

Taken from [?].

Relationship, not Position

- In an array each item occupies a particular position.
  - This position can be directly accessed using an index number.
    - It is like a row of houses:
    - you can find a particular house using its address.
- In a list the only way to find a particular element is to follow along the chain of elements.
  - It is more like human relations.
    - Maybe you ask Harry where Bob is.
    - Harry does not know, but he thinks Jane might know,
    - so you go and ask Jane...

Taken from [?].

Linked lists

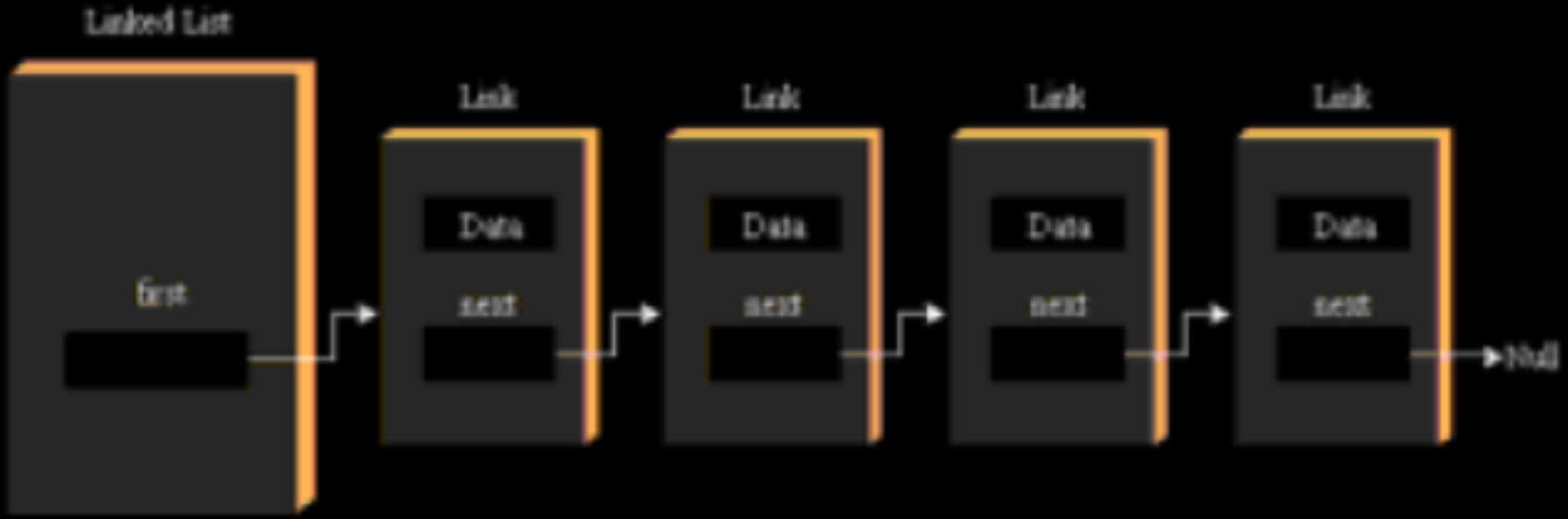

Figure: Links in a list. Taken from [?].

Linked lists in memory

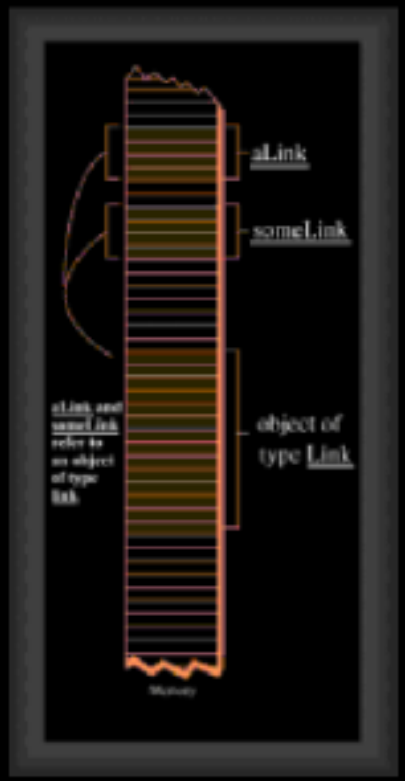

Figure: Links and references in memory. Taken from [?].

Simulator of a Linked list

http://visualgo.net/list.html

Insert a new node

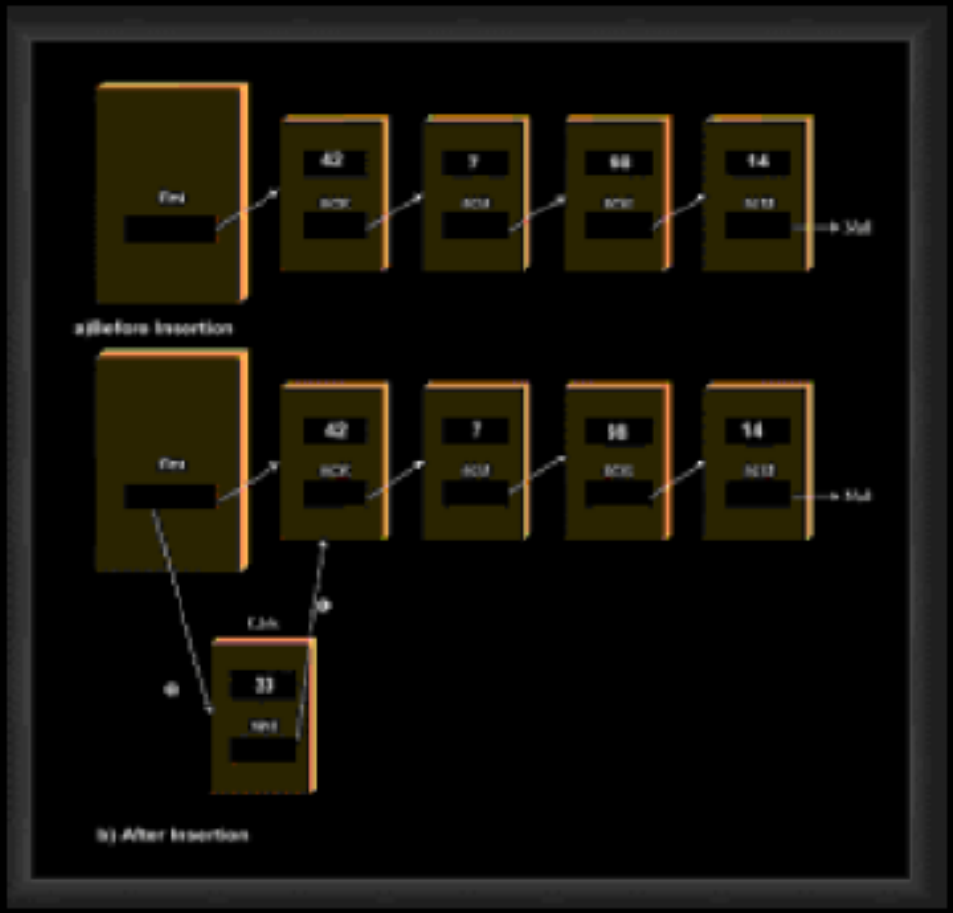

Figure: Inserting a new link. Taken from [?].

Delete a link

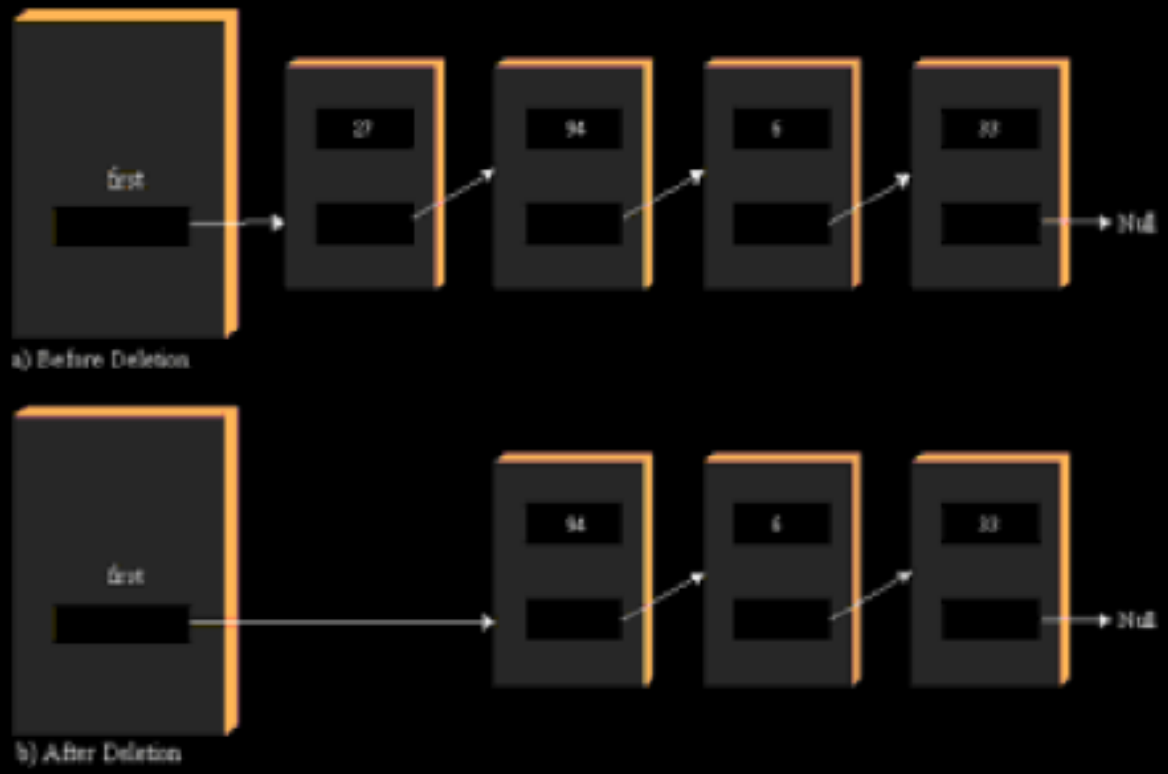

Figure: Deleting a link. Taken from [?].

Doubly-linked lists

- With ordinary linked lists is that it is difficult to traverse backward along the list.
- For example, imagine a text editor in which a linked list is used to store the text.
  - Each text line on the screen is stored as a String object embedded in a link.
  - When the editor's user moves the cursor downward on the screen, the program steps to the next link to manipulate or display the new line.
  - But what happens if the user moves the cursor upward?

Taken from [?].

Doubly-linked lists

- With ordinary linked lists is that it is difficult to traverse backward along the list.
- For example, imagine a text editor in which a linked list is used to store the text.
  - Each text line on the screen is stored as a String object embedded in a link.
  - When the editor's user moves the cursor downward on the screen, the program steps to the next link to manipulate or display the new line.
  - But what happens if the user moves the cursor upward?

Taken from [?].

Doubly-linked lists

- With ordinary linked lists is that it is difficult to traverse backward along the list.
- For example, imagine a text editor in which a linked list is used to store the text.
  - Each text line on the screen is stored as a String object embedded in a link.
  - When the editor's user moves the cursor downward on the screen, the program steps to the next link to manipulate or display the new line.
  - But what happens if the user moves the cursor upward?

Taken from [?].

Doubly-linked lists (2)

Figure: A doubly-linked list. Taken from [?].

Doubly-linked lists (3)

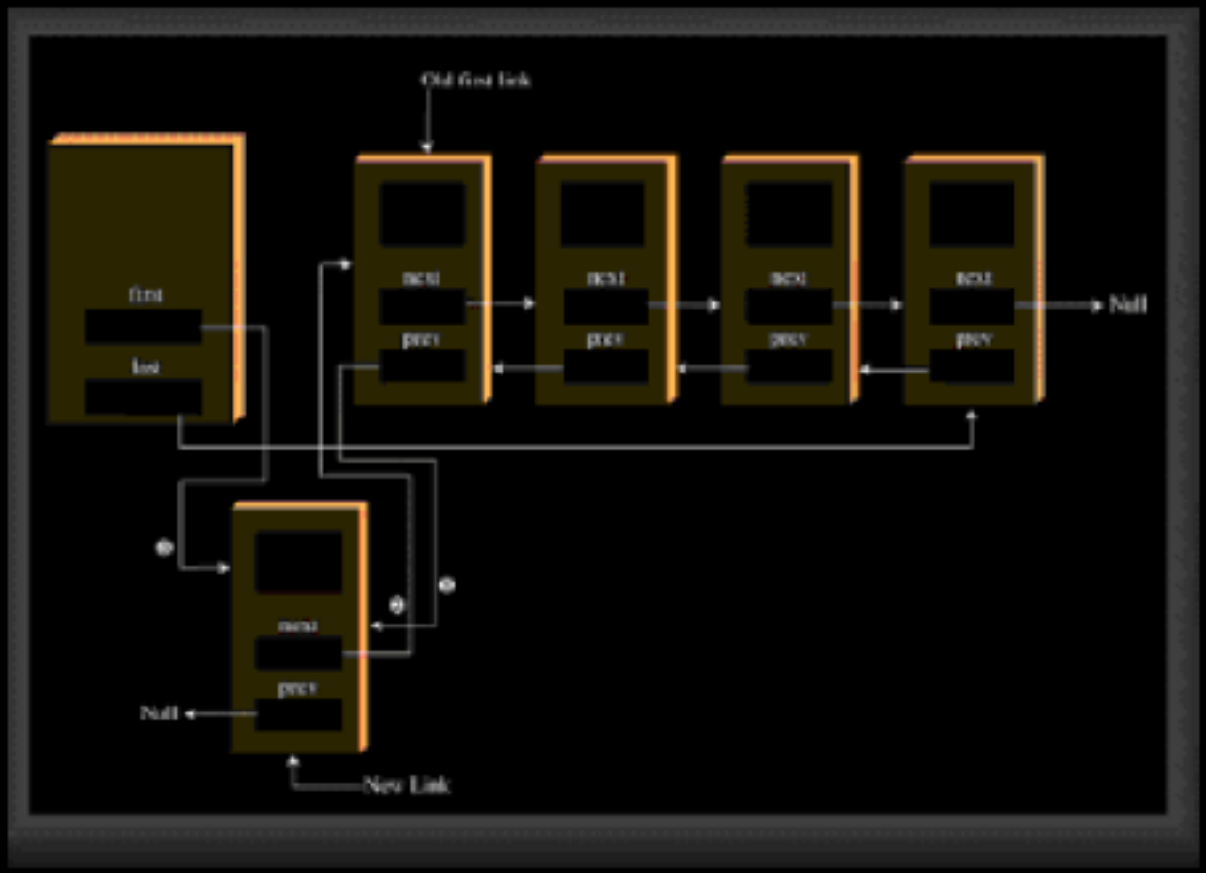

Figure: Insertion at the beginning. Taken from [?].

Doubly-linked lists (4)

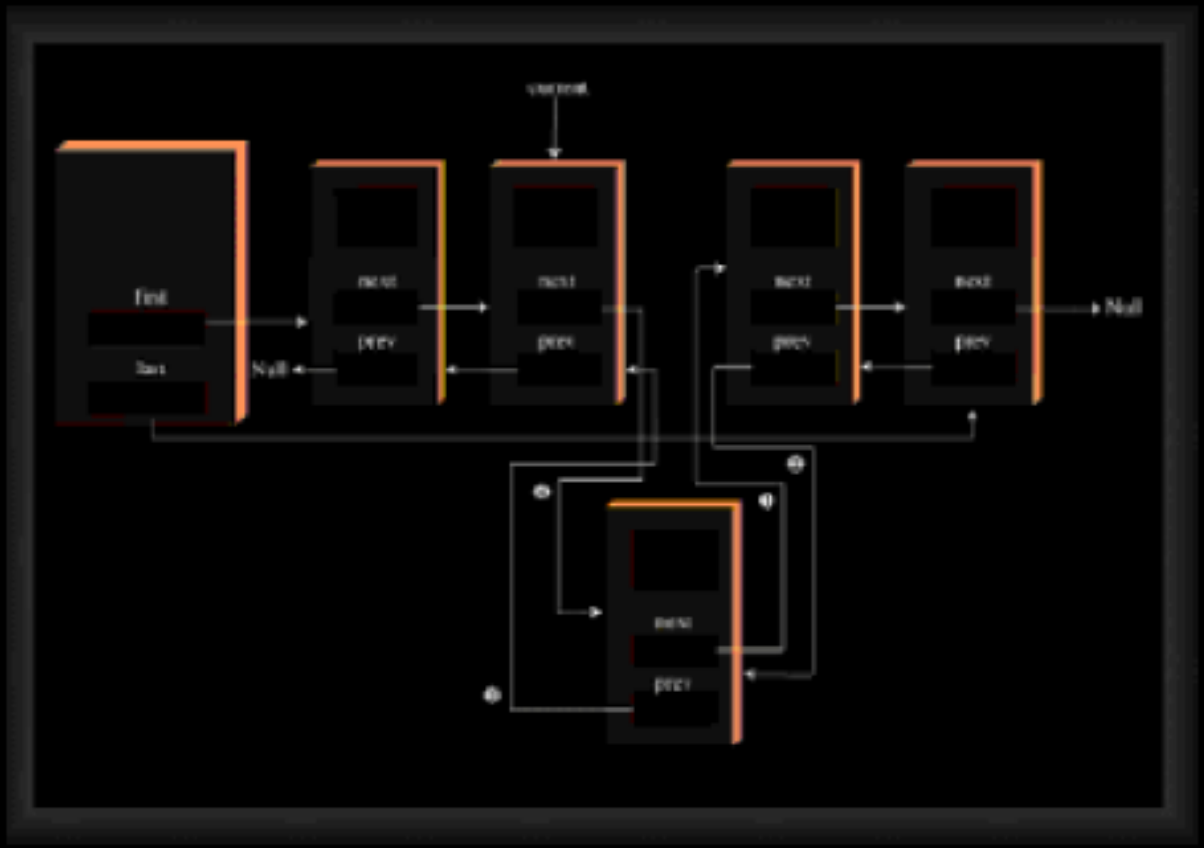

Figure: Insertion at an arbitrary location. Taken from [?].

Doubly-linked lists (5)

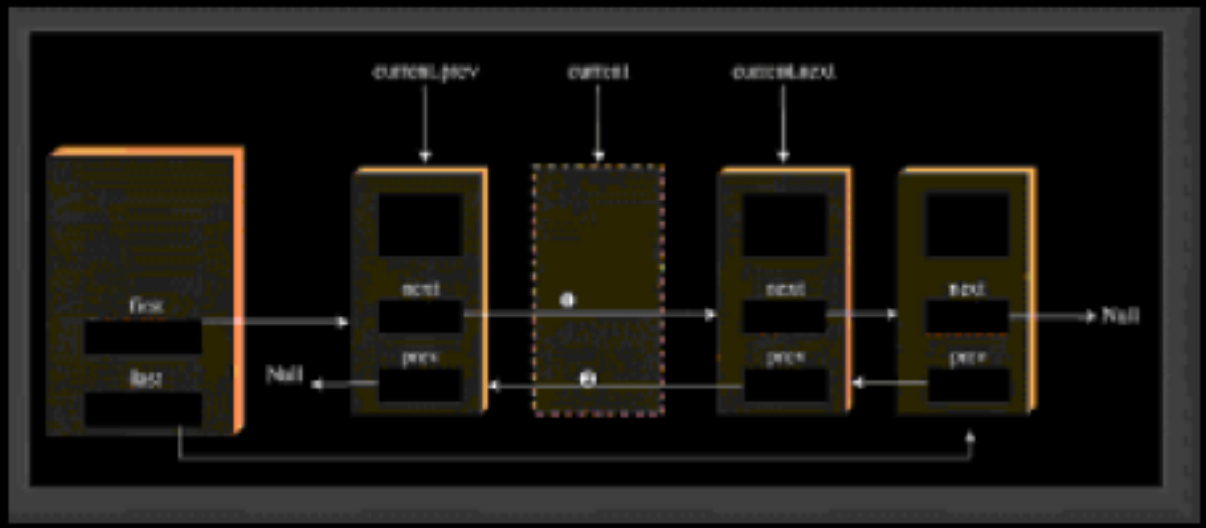

Figure: Deleting an arbitrary link. Taken from [?].

Quiz questions

- Insertion in an array is slow ($O(n)$); insertion in a linked list is fast ($O(1)$).
- Random access in an array is fast ($O(1)$); random access in a linked list is slow ($O(n)$).
- Backward traversal of Singly-linked lists is slow ($O(n^2)$; for Doubly-linked lists and arrays is fast ($O(n)$).

Quiz questions

- Insertion in an array is slow ($O(n)$); insertion in a linked list is fast ($O(1)$).
- Random access in an array is fast ($O(1)$); random access in a linked list is slow ($O(n)$).
- Backward traversal of Singly-linked lists is slow ($O(n^2)$; for Doubly-linked lists and arrays is fast ($O(n)$).

Quiz questions

- Insertion in an array is slow ($O(n)$); insertion in a linked list is fast ($O(1)$).
- Random access in an array is fast ($O(1)$); random access in a linked list is slow ($O(n)$).
- Backward traversal of Singly-linked lists is slow ($O(n^2)$; for Doubly-linked lists and arrays is fast ($O(n)$).

References

- Please learn how to reference images, trademarks, videos and fragments of code.
- Avoid plagiarism

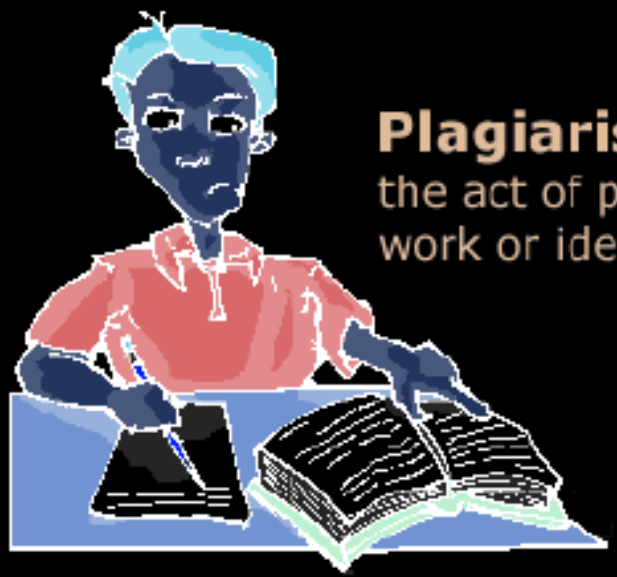

Figure: Figure about plagiarism, University of Malta [?]

References

Further reading

- Lists
  - Jorge Villalobos. Introducción a las Estructuras de Datos: Aprendizaje Activo Basado en Casos. Nivel 3. Páginas 177 – 209.